Stanford Exploration Project, Report 72, December 18, 1997, pages 1–176

Stanford Exploration Project, Report 72, December 18, 1997, pages 1–176

Finite difference calculation of Green's functions

Lin Zhang[1]

keywords: *algorithm, traveltime, amplitudes, ray tracing, modeling*

ABSTRACT

With the asymptotic ray approximation, the Green's functions of the elastic wave equation are characterized by traveltime and amplitude functions. A new finite difference algorithm is developed to calculate the traveltime and amplitude functions of the first arrival seismic waves for a given 2-D medium. This algorithm combines features of previous algorithms. To achieve efficiency and accuracy, the partial differential equations are solved in polar coordinates. The resulting codes are fully vectorizable. The stability of the algorithm is ensured by adapting the step sizes of the wavefront extrapolation to the local directions of wave propagation. For traveltime calculations, the algorithm handles slowness models containing large contrast. For amplitude calculations, the algorithm handles both line and point source geometries, but requires slowness models to be smooth since transmission losses are not taken into account. The source radiation patterns, if known, can also be incorporated into the amplitude calculation. Examples with different types of slowness models show that the method works. A potential application of the method and its extension to 3-D could be useful for Kirchhoff prestack depth migration.

INTRODUCTION

The largest portion of the computational effort in migrating data in heterogeneous media with an integral method is the calculation of the required Green's functions. With asymptotic ray theory, the approximate Green's functions for compressional waves and shear waves can be decoupled (Červený et al., 1977). Each Greens' function is characterized by a traveltime function and an amplitude function. Traditionally, these traveltime and amplitude functions are obtained by performing dynamic ray tracing to compute the traveltimes and amplitudes along rays, and then interpolating them onto a regular grid (Červený,1987; Beydoun and Keho, 1987). However, for complicated slowness models, the interpolation process may be theoretically invalid and computationally expensive.

[1]**email:** not available

Vidale (1988) developed a finite difference method that computes the traveltimes directly on a regular grid in Cartesian coordinates. The method can be interpreted as a wavefront extrapolation process that starts from the source location and proceeds on an expanding square box. Van Trier and Symes (1991) described a slightly different algorithm in which the gradients of the traveltimes, instead of the traveltimes themselves, are extrapolated. Their method conducts the calculation in polar coordinates and transforms the results to the Cartesian coordinates afterwards. Van Trier and Symes showed that their method often achieves better efficiency and accuracy than Vidale's method. However, the results of many numerical experiments show that both algorithms encounter serious stability problems when applied to models containing large slowness discontinuities.

Podvin and Lecomte (1991) reported several significant improvements on Vidale's method. First, their algorithm systematically applies Huygens' principle in the finite difference approximation so that the method provides accurate traveltimes in the presence of extremely severe, arbitrarily shaped slowness contrasts. Second, they developed a massively parallel implementation of the algorithm. In this implementation, each grid point is associated with a processor. All the processors simultaneously update the traveltimes at the associated grid points, which is like a relaxation process. Third, to ensure the results of the sequential implementation are minimum traveltimes, Podvin and Lecomte introduced backward extrapolation. This procedure captures the first arrivals that propagate into the square computational box.

While acknowledging the promising results given by Podvin and Lecomte, I find three weaknesses in their algorithm, for which better solutions may be explored. First the massive parallel implementation does not effectively utilize computational power because, during the computation, most of the processors away from the first arrival wavefront perform dummy operations. In 3-D, the computation involved in each processor is quite complicated. Second, similar to Vidale's algorithm, the sequential implementation can not be vectorized. Third, the backward extrapolation introduces extra cost and should be avoided whenever possible.

I have developed a new traveltime calculation method that combines the advantages of the existing algorithms. I use a simplified finite difference representation of Podvin and Lecomte's scheme so that the new algorithm works for large contrast slowness models. I choose to work in polar coordinates to make the algorithm fully vectorizable. This choice also avoids performing backward extrapolation if one assumes that waves seldom propagate back toward the source. The stability of the algorithm is ensured by adapting the step size of the extrapolation to the directions of the wave propagation. The algorithm can also be used to solve the partial differential equations associated with the amplitude calculation. The resulted algorithm is relatively more efficient than Vidale's method (1990). The cost of adding the amplitude calculation to the algorithm is about twice the cost of calculating the traveltimes only.

This paper is organized into three parts. The first part explains the required theory. I begin with a brief review of some well-known results in the asymptotic ray theory. Then, I address the theoretical aspects of the traveltime and amplitude calculations. In the second part, I describe the implementational details of the finite difference algorithm. The third part of the paper demonstrates the method with several examples. Finally, I conclude the paper by discussing the applications of the method.

THEORY

The elastic wave equation for an isotropic, 2-dimensional medium has a well-known form:

$$\begin{aligned}
\rho \frac{\partial^2 u_1}{\partial t^2} &= \frac{\partial}{\partial x}\left[\lambda(\frac{\partial u_1}{\partial x} + \frac{\partial u_2}{\partial z}) + 2\mu \frac{\partial u_1}{\partial x}\right] + \frac{\partial}{\partial z}\left[\mu(\frac{\partial u_1}{\partial z} + \frac{\partial u_2}{\partial x})\right] \\
\rho \frac{\partial^2 u_2}{\partial t^2} &= \frac{\partial}{\partial z}\left[\lambda(\frac{\partial u_1}{\partial x} + \frac{\partial u_2}{\partial z}) + 2\mu \frac{\partial u_2}{\partial z}\right] + \frac{\partial}{\partial x}\left[\mu(\frac{\partial u_2}{\partial x} + \frac{\partial u_1}{\partial z})\right],
\end{aligned} \tag{1}$$

where u_1 and u_2 are the x- and z-components of the displacement vector, $\lambda = \lambda(x, z)$ and $\mu = \mu(x, z)$ are the elastic constants of the media, and ρ is the density of the medium. Using asymptotic ray theory, we can expand the Green's function of this wave equation into a ray series at high frequencies (Červený et al., 1977). The leading term of this ray series has two components: G_P for the compressional wave (P-wave) and G_S for the shear wave (S-wave):

$$G_c(\omega, x, z; x_0, z_0) = A_c(x, z; x_0, z_0) e^{i\omega\tau_c(x,z;x_0,z_0)}. \quad c = P \text{ or } S \tag{2}$$

In this equation, τ_c is the traveltime it takes for a wave to propagate from a source location (x_0, z_0) to an observation location (x, z). The traveltime function satisfies the eikonal equation of wave equation (1). If we assume that the medium is smoothly varying, then function A_c describes the amplitude decay of this propagation due to geometric spreading, and it satisfies the first order transport equation. Because the Green's functions for P-wave and S-wave have a common form, from now on, we drop the subscripts P and S, and make distinctions only when necessary.

Traveltimes

The eikonal equation in two dimensions relates the two gradient components of the traveltime field as follows:

$$\tau_x^2(x, z) + \tau_z^2(x, z) = m^2(x, z), \tag{3}$$

where subscripts x and z denote partial derivatives with respect to x and z, respectively, and $m(x,z)$ is the slowness model of the medium determined by

$$m(x,z) = \begin{cases} \sqrt{\dfrac{\rho(x,z)}{\lambda(x,z)+2\mu(x,z)}} & \text{for } P\text{-waves} \\ \\ \sqrt{\dfrac{\rho(x,z)}{\mu(x,z)}} & \text{for } S\text{-waves.} \end{cases} \tag{4}$$

Traditionally, the eikonal equation is solved by using the method of characteristics. The characteristic curves of this equation are rays described by the ray equations

$$\begin{cases} \dfrac{dx}{ds} = \dfrac{\tau_x(x,z)}{m(x,z)} & \dfrac{d\tau_x}{ds} = m_x(x,z) \\ \\ \dfrac{dz}{ds} = \dfrac{\tau_z(x,z)}{m(x,z)}. & \dfrac{d\tau_z}{ds} = m_z(x,z), \end{cases} \tag{5}$$

where variable s is the arc-length of the ray. The solutions to the ray equation define a transformation between the coordinate system (x,z) and a new coordinate system (s,γ), named ray coordinates, as follows:

$$\begin{cases} x = \hat{x}(s,\gamma) \\ \\ z = \hat{z}(s,\gamma) \end{cases} \tag{6}$$

For each fixed γ, this pair of equations give the Cartesian coordinates of a ray. The variations of γ span a set of rays covering the whole (x,z) space. If we substitute (x,z) in equation (6) into the eikonal equation, we obtain an ordinary differential equation (ODE) along each ray:

$$\frac{d\tau}{ds} = m(\hat{x}(s,\gamma), \hat{z}(s,\gamma)). \tag{7}$$

The solution of this ODE is simply an integral along the ray:

$$\tau(\hat{x}(s,\gamma), \hat{z}(s,\gamma)) = \tau(\hat{x}(s_0,\gamma), \hat{z}(s_0,\gamma)) + \int_{s_0}^{s} m(\hat{x}(\xi,\gamma), \hat{z}(\xi,\gamma))d\xi. \tag{8}$$

This traveltime calculation method is called ray tracing method. The ray tracing method computes traveltimes along rays, hence generates a traveltime map on an irregular grid.

Many seismic modeling and imaging algorithms require a traveltime map on a regular grid. An efficient way to obtain this traveltime map is to solve the eikonal equation directly using a finite difference scheme. The finite difference calculation of traveltimes is actually an extrapolation process. The traveltime function is locally extrapolated from the grid points where the traveltimes are known to their neighboring grid points where the traveltimes are unknown by following the principle that the gradients of the traveltimes satisfy the eikonal equation. This process starts from the source position where the traveltime is zero, and is repeated until all grid points are filled. We can also extrapolate the gradients of traveltimes. The gradients of traveltimes are vectors and have two components in two dimensions. In order to determine both components, we need one more equation besides the eikonal equation. Van Trier and Symes (1990) use the following one:

$$\frac{\partial \tau_z}{\partial x} = \frac{\partial \tau_x}{\partial z}, \tag{9}$$

where (τ_x, τ_z) are the two components of the traveltime gradient. Van Trier and Symes interpret this equation using the flux conservation principle. If we rewrite equation (9) as $\nabla \times \nabla\tau = 0$, we see that this equation states that a gradient field is curl-free. Once the traveltime gradients are computed, their integrations give the traveltimes.

Amplitudes

Collecting the leading terms of amplitude in the ray series expansion yields the zeroth order transport equation (Červený et al., 1977):

$$\nabla\tau \cdot \nabla A + \frac{1}{2}\left[\nabla \ln\rho \cdot \nabla\tau + \nabla^2\tau\right] A = 0. \tag{10}$$

If we assume that the traveltime function has been found by solving the eikonal equation, then equation (10) is a first order, linear partial differential equation (PDE). We see that equation (10) contains the term $\nabla^2\tau$ that is related to the derivatives of slowness function with respect to the spatial coordinates. Thus, the validity of this equation requires the slowness function to be differentiable. If this requirement is met, we can presumably solve this equation using a finite difference method. An easier approach (Červený et al., 1977) is to solve the transport equation in ray coordinates and transform the analytical solution of the resulting ODE to Cartesian coordinates, which yields

$$A(x,z) = R(\gamma(x,z))\sqrt{\frac{\rho(x_0,z_0)m(x,z)}{|J(x,z)|\rho(x,z)m(x_0,z_0)}}, \tag{11}$$

where R is the radiation pattern of the source, $\gamma(x,z)$ is the take-off angle of the ray that reaches point (x,z), $J(x,z)$ is the Jacobian of the transformation from Cartesian coordinates to ray coordinates.

$$J(x,z) = \begin{pmatrix} \frac{\partial x}{\partial s} & \frac{\partial z}{\partial s} \\ \frac{\partial x}{\partial \gamma} & \frac{\partial z}{\partial \gamma} \end{pmatrix}. \tag{12}$$

In Appendix A, I show that for a 2-D medium and a line source (the line source is aligned along the third dimension on which the slowness does not depend),

$$|J(x,z)| = \frac{1}{\sqrt{\left(\frac{\partial\gamma}{\partial x}\right)^2 + \left(\frac{\partial\gamma}{\partial z}\right)^2}}. \tag{13}$$

Also in Appendix A, I show that function $\gamma(x,z)$ can be computed by solving the first order linear PDE

$$\tau_x\gamma_x + \tau_z\gamma_z = 0. \tag{14}$$

with the initial condition that $\gamma(x,z)$ is equal to the take-off angle at the source position.

In seismic exploration, point sources may be more appropriate. Cerveny (1981) shows that for point source, function J for line source should be multiplied by a factor η to take account the geometric spreading in the third dimension. He showed that

$$\eta(\hat{x}(s,\gamma),\hat{z}(s,\gamma)) = m(x_0,z_0)\int_{s_0}^{s}\frac{1}{m(\hat{x}(\xi,\gamma),\hat{z}(\xi,\gamma))}d\xi \tag{15}$$

where (x_0,z_0) is the coordinate of the source position. I show, in Appendix B, that function $\eta(x,z)$ can be calculated by solving a first order linear PDE:

$$\tau_x\eta_x + \tau_z\eta_z = m(x_0,z_0), \tag{16}$$

The initial condition for solving equation (16) is $\eta(x_0,z_0)\rightarrow 0$. The algorithm used for solving the eikonal equation, after some modifications, can be used to solve PDEs (14) and (16). It should be noted that all these formulas are valid only in a regular region in which no caustics exist.

ALGORITHM

Stability, accuracy and efficiency are three utmost concerns of a finite difference algorithm. The finite difference calculation of traveltimes is an extrapolation process. To maintain the stability of the calculation, the extrapolation must be causal. For this reason, Vidale's algorithm (1988) starts the extrapolation from the so called "relative minimum points". However, as pointed out by Van Trier and Symes (1991), this ordering procedure complicates the algorithm and makes it non-vectorizable. Consequently, I follow Van Trier and Symes' approach and do the computation in polar coordinates. Stability is ensured by adapting the step size of extrapolation to the local directions of wave propagation.

Slowness model

To handle a slowness model containing large contrasts, my algorithm uses a simplified finite difference version of Podvin and Lecomte (1991). Therefore, the true slowness model is approximated by a discrete slowness model that consists of constant slowness cells in polar coordinates. I assume that the grid sizes are small so that each individual cell can be treated as a small rectangle of Δr by $r\Delta\theta$. At the boundaries of each cell, the slowness function is generally discontinuous. The interfaces of the slowness model are formed along the boundaries of the cells. As we will see later in the examples, this approximation is adequate for the traveltime calculation, but inadequate for the amplitude calculation when slowness varies sharply.

Computation of traveltimes

A good finite difference scheme should incorporate the relevant physical law into its representation. Extrapolating traveltimes is like propagating wavefronts; Huygens' principle and Fermat's principle should be respected. Extrapolating the gradients of traveltimes is like extending rays; Snell's law should be respected. Because these two extrapolations simulate the same physical phenomenon, the algorithm to do one can be translated and used to do the other. The algorithm described below does two extrapolations simultaneously.

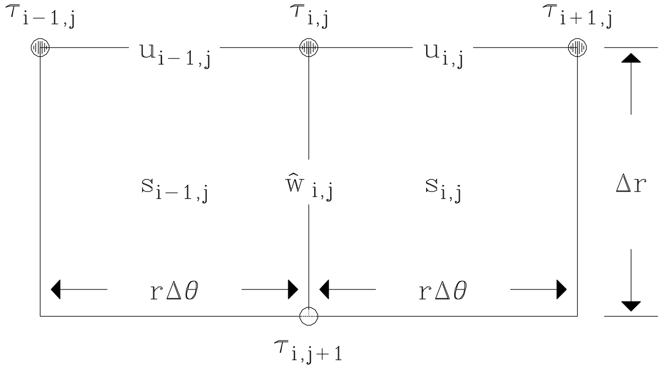

Figure 1: The grid and notations. lin1-vfdtime1 [ER]

The eikonal equation in polar coordinates is as follows:

$$\tau_r^2 + \frac{1}{r^2}\tau_\theta^2 = m^2(\theta, r), \tag{17}$$

where subscripts r and θ denote the partial derivatives with respect to r and θ, respectively. The initial condition is

$$\tau(\theta, r) = m(\theta, r)r \qquad \text{for } r \leq r_0,$$

where r_0 defines a circle centered at the source point, within which the slowness is constant. Starting from this circle, we extrapolate the traveltime field in the radial directions. As indicated in Figure **??**, I use $\tau_{i,j}$ to denote the traveltime at grid point (θ_i, r_j). Then, in the jth step of the traveltime extrapolation, we know the traveltime $\tau_{i,j}$ for all i and we want to find the traveltime $\tau_{i,j+1}$ for all i.

According to Snell's law, the tangential component of the traveltime gradient to a boundary is continuous across the boundary. I define $u_{i,j}$ to be the tangential component of the traveltime gradient to the constant radius boundary. If we approximate the wavefront within a cell with a local plane wave, then we have

$$u_{i,j} = \frac{1}{r_j}\frac{\partial \tau}{\partial \theta} = \frac{\tau_{i+1,j} - \tau_{i,j}}{r_j \Delta\theta}. \tag{18}$$

The orthogonal component of the traveltime gradient can be computed from the eikonal equation as follows:

$$w_{i,j} = \sqrt{s_{i,j}^2 - u_{i,j}^2}. \tag{19}$$

These two components determine the propagation direction of the local plane wave. If we define

$$\hat{w}_{i,j} = \frac{\tau_{i,j+1} - \tau_{i,j}}{\Delta r} \tag{20}$$

then we can use the principles of Podvin and Lecomte's scheme (1991) to calculate $\hat{w}_{i,j}$ in the following four cases:

1. $u_{i-1,j} \geq 0$ and $u_{i,j} \leq 0$

$$\hat{w}_{i,j} = \min(w_{i-1,j}, w_{i,j}, s_{i-1,j}, s_{i,j})$$

2. $u_{i-1,j} \geq 0$ and $u_{i,j} > 0$

$$\hat{w}_{i,j} = \min(w_{i-1,j}, s_{i,j})$$

3. $u_{i-1,j} < 0$ and $u_{i,j} \leq 0$

$$\hat{w}_{i,j} = \min(w_{i,j}, s_{i-1,j})$$

4. $u_{i-1,j} < 0$ and $u_{i,j} > 0$

$$\hat{w}_{i,j} = \min(s_{i-1,j}, s_{i,j}).$$

The appearance of slowness in these formulas indicates that the possible refractions along the constant θ boundary are taken into account. The "min" function ensures that case gives the minimum traveltime. The causality of the extrapolation requires that

$$w_{i-1,j} > s_{i-1,j}\frac{\Delta r}{\sqrt{\Delta r^2 + (r_j\theta)^2}} \qquad \text{and} \qquad w_{i,j} > s_{i,j}\frac{\Delta r}{\sqrt{\Delta r^2 + (r_j\theta)^2}}. \tag{21}$$

If we adapt the step size Δr of the extrapolation to the propagation direction of the local plane wave and choose

$$\Delta r = \min_{\text{all } i}[r_j\Delta\theta \left|\frac{w_{i,j}}{u_{i,j}}\right|], \tag{22}$$

then the inequalities in equation (21) are guaranteed. Solving equation (20) for $t_{i,j+1}$ gives

$$t_{i,j+1} = t_{i,j} + \hat{w}_{i,j}\Delta r \qquad \text{for all } i, \tag{23}$$

which completes one step of the extrapolation.

The implementation of this algorithm is quite simple and the program is fully vectorizable. For example, the computation of $\hat{w}_{i,j}$ for all i can be coded as follows:

```
do i = 1,n
   sign1 = 0.5*(1.+sign(1.,u(i-1)))
   sign2 = 0.5*(1.+sign(1.,-u(i)))
   hatw1 = sign1*min(w(i-1),s(i))+(1.-sign1)*s(i-1)
   hatw2 = sign2*min(w(i),s(i-1))+(1.-sign2)*s(i)
   hatw(i) = min(hatw1,hatw2)
enddo
```

One problem that has not been addressed yet is the possibility of refractions along the constant radius boundaries, which may initiate wave propagations toward the source. In practice, such situation seldom happen. To make the algorithm more complete, I handle such situation by adopting Podvin and Lecomte's method in polar coordinates and performing backward extrapolation as necessary. After obtaining the traveltimes in polar coordinates, I map them to Cartesian coordinates through interpolation.

Computation of amplitudes

To compute the amplitudes, we need to solve equations (14) and (16). In polar coordinates, these two equations become:

$$\tau_r \gamma_r + \frac{1}{r^2} \tau_\theta \gamma_\theta = 0 \tag{24}$$

and

$$\tau_r \eta_r + \frac{1}{r^2} \tau_\theta \eta_\theta = m(x_0, z_0), \tag{25}$$

respectively. Because the traveltimes are known now, these two equations are first-order linear partial differential equations in γ and η. The algorithms for solving these two equations are very similar to the one for traveltime calculation. To solve equation (24), we replace equations (18) to (20) by

$$f_{i,j} = \frac{\gamma_{i+1,j} - \gamma_{i,j}}{r_j \Delta\theta}, \tag{26}$$

$$g_{i,j} = -\frac{u_{i,j}}{w_{i,j}} f_{i,j} \tag{27}$$

and

$$\hat{g}_{i,j} = \frac{\gamma_{i,j+1} - \gamma_{i,j}}{\Delta r}, \tag{28}$$

respectively. As in the calculation of $\hat{w}_{i,j}$, the calculation of $\hat{g}_{i,j}$ also involves four different cases:

1. $u_{i-1,j} \geq 0$ and $u_{i,j} \leq 0$

$$\hat{g}_{i,j} = \begin{cases} g_{i-1,j} & \text{if } \hat{w}_{i,j} = w_{i-1,j} \\ g_{i,j} & \text{if } \hat{w}_{i,j} = w_{i,j} \\ 0 & \text{if } \hat{w}_{i,j} = s_{i-1,j} \text{ or } s_{i,j} \end{cases}$$

2. $u_{i-1,j} \geq 0$ and $u_{i,j} > 0$

$$\hat{g}_{i,j} = \begin{cases} g_{i-1,j} & \text{if } \hat{w}_{i,j} = w_{i-1,j} \\ 0 & \text{if } \hat{w}_{i,j} = s_{i,j} \end{cases}$$

3. $u_{i-1,j} < 0$ and $u_{i,j} \le 0$

$$\hat{g}_{i,j} = \begin{cases} g_{i,j} & \text{if } \hat{w}_{i,j} = w_{i,j} \\ 0 & \text{if } \hat{w}_{i,j} = s_{i-1,j} \end{cases}$$

4. $u_{i-1,j} < 0$ and $u_{i,j} > 0$

$$\hat{g}_{i,j} = 0$$

If $\hat{w}_{i,j}$ and $\hat{g}_{i,j}$ are calculated in one subroutine, then the implementation is simple and the program is fully vectorizable, which is shown as follows:

```
do i = 1,n+1
   sign1 = 0.5*(1.+sign(1.,u(i-1)))
   sign2 = 0.5*(1.+sign(1.,-u(i)))
   hatw1 = sign1*min(w(i-1),s(i))+(1.-sign1)*s(i-1)
   hatw2 = sign2*min(w(i),s(i-1))+(1.-sign2)*s(i)
   hatw(i) = min(hatw1,hatw2)
   sign3 = 0.5*(1.+sign(1.,w(i-1)-s(i)))
   sign4 = 0.5*(1.+sign(1.,w(i)-s(i-1)))
   sign5 = 0.5*(1.+sign(1.,hatw2-hatw1))
   hatg1 = sign1*(1.-sign3)*g(i-1)
   hatg2 = sign2*(1.-sign4)*g(i)
   hatg(i) = sign5*hatg1+(1.-sign5)*hatg2
enddo
```

From equation (28), we have

$$\gamma_{i,j+1} = \gamma_{i,j} + \hat{g}_{i,j}\Delta r. \tag{29}$$

I can apply Podvin and Lecomte's method to the computation of function $\gamma(r,\theta)$ so that backward extrapolation can be handled.

To solve equation (25), I use

$$\eta_r = \frac{1}{\tau_r}[m(x_0,z_0) - \frac{1}{r^2}\tau_\theta\eta_\theta]$$

to extrapolate the function $\eta(r,\theta)$. Referring to the algorithm for calculating $\tau(r,\theta)$ and $\gamma(r,\theta)$, one can easily construct the algorithm for calculating function $\eta(r,\theta)$.

The Jacobian function in polar coordinates can be determined from (13) as follows:

$$|J(r,\theta)| = \frac{1}{\sqrt{\left(\frac{\partial\gamma}{\partial r}\right)^2 + \left(\frac{\partial\gamma}{r\partial\theta}\right)^2}}. \tag{30}$$

We can either map this Jacobian function to Cartesian coordinates and use equation (11) to calculate the amplitudes, or calculate the amplitudes in polar coordinates and then map them to Cartesian coordinates.

EXAMPLES

In this section, I show examples for both traveltime and amplitude calculations.

Traveltimes and rays

The traveltime calculation algorithm can handle virtually arbitrary slowness models. Figure **??** shows the results for a medium in which velocity linearly increases with depth. The velocity gradient is 1.5 1/sec. The background of this figure shows the slowness model. I plot contour lines of the traveltimes on this background. The contour lines of the traveltimes are the trajectories of the wavefronts. We see that the wavefronts are stretched downward because the slowness decreases with depth. Because the function $\gamma(x, z)$ defined in equation (14) is constant along each ray, I plot the contour lines of this function to show the rays. The values of the contours are uniformly spaced so that the rays plotted are shot with uniform take-off angles. As expected, we see many overturned rays.

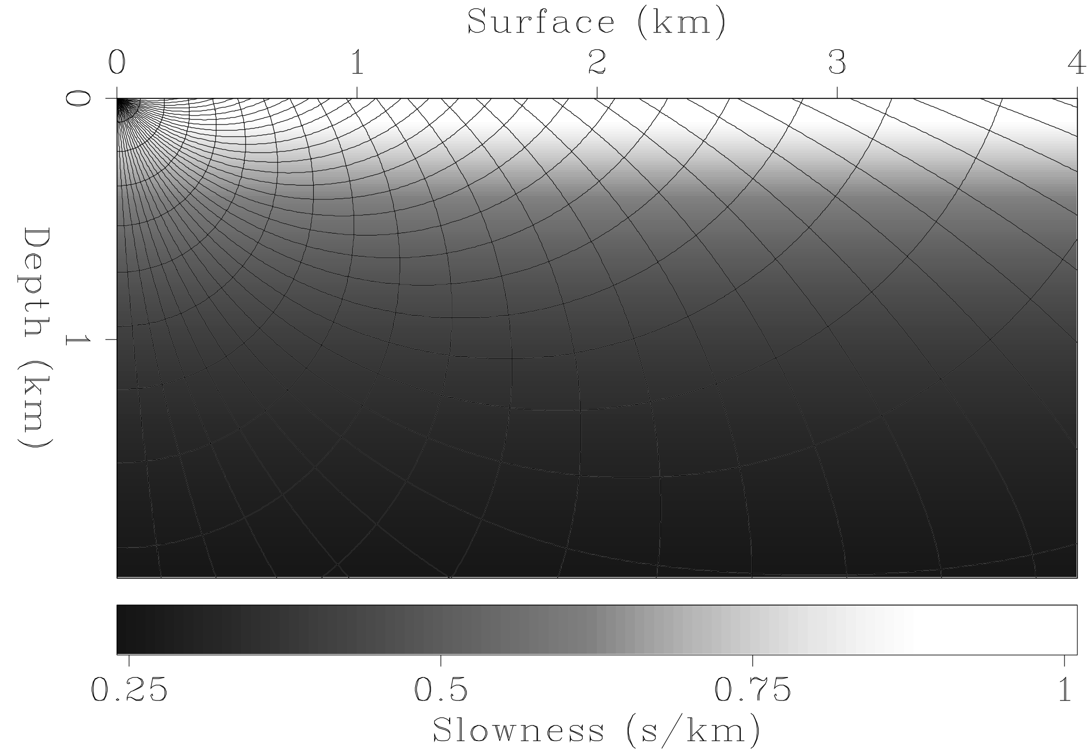

Figure 2: Wavefronts and rays for a model with a velocity function that is linearly increasing as a function of depth. The background shows the slowness model. The source is at the upper left corner of the figure. The traveltime contour lines are drawn at 0.1 second intervals. $\boxed{\text{lin1-ttmraygra}}$ [ER]

The slowness model in the second example is a medium with two layers. The velocity contrast between the top layer and the bottom layer is 1 to 2. Figure **??**

shows the results of the calculations. The content of this figure is similar to Figure **??**. We see the first arrivals associated with both transmitted waves and refracted waves. We see the transmitted rays but not the rays refracted along the interface. This is because all the refracted rays originate from a single ray; therefore the values of the function $\gamma(x, z)$ along all the refracted rays are equal. The thick curve that appears in the top layer is the boundary that separates the first arrivals from the transmitted waves and the refracted waves.

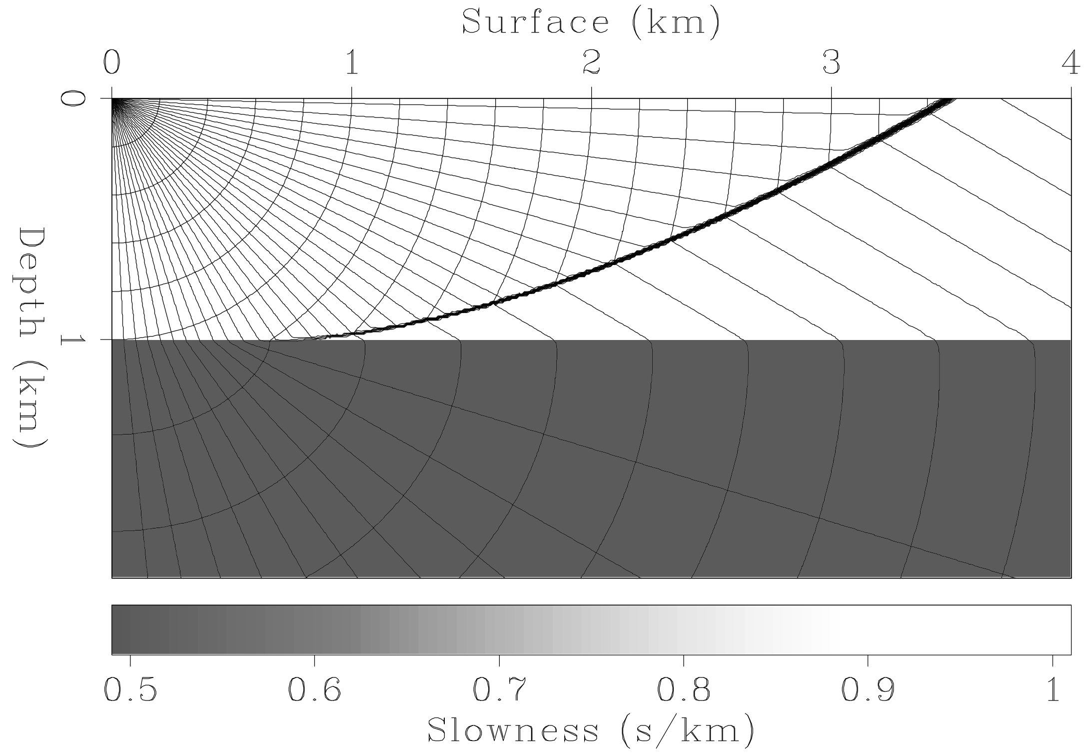

Figure 3: Wavefronts and rays for a model with two constant-velocity layers. The background shows the slowness model. The source is at the upper left corner of the figure. The traveltime contour lines are drawn at 0.2 second intervals. $\boxed{\text{lin1-ttmraylay}}$ [ER]

The final example for the traveltime calculation uses a slowness model containing an anomaly with large velocity contrast. The results of the calculations are shown in Figure **??**. The looping contour lines within the anomaly are due to the finite grid sizes. Clearly, in the lower right part of the anomaly, the first arrivals propagate back toward the source. This part is calculated by backward extrapolation.

Amplitudes

The amplitude calculation requires that the slowness models are smooth. I apply the algorithm to compute the amplitudes for a medium having a velocity function that increases linearly with depth. The velocity gradient is 1.5 1/sec. Figure **??** shows

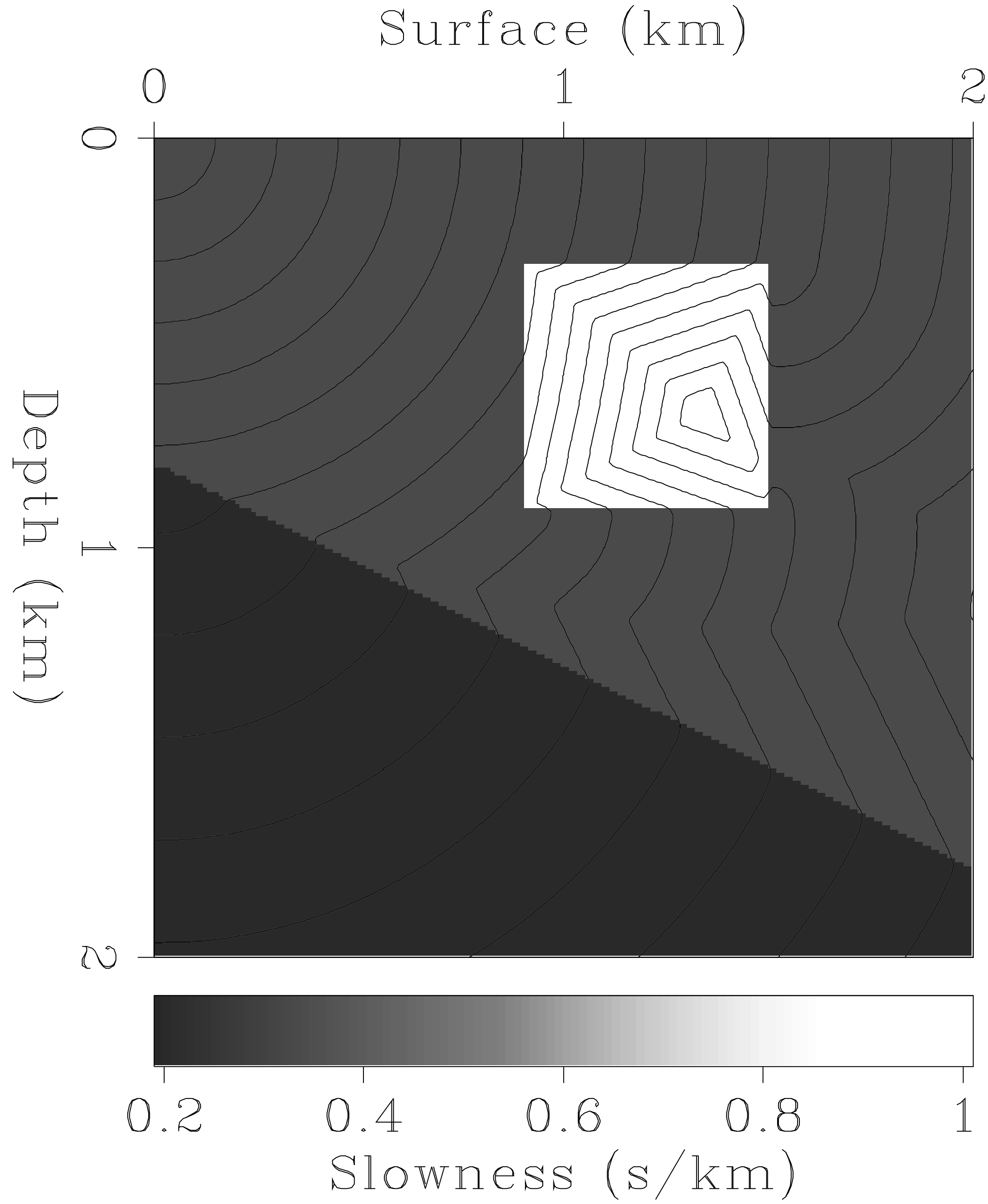

Figure 4: Wavefronts for a model with a sharply-contrasted anomaly. The background shows the slowness model. The source is at the upper left corner of the figure. The traveltime contour lines are drawn at 0.05 second intervals. lin1-ttmrayrec [ER]

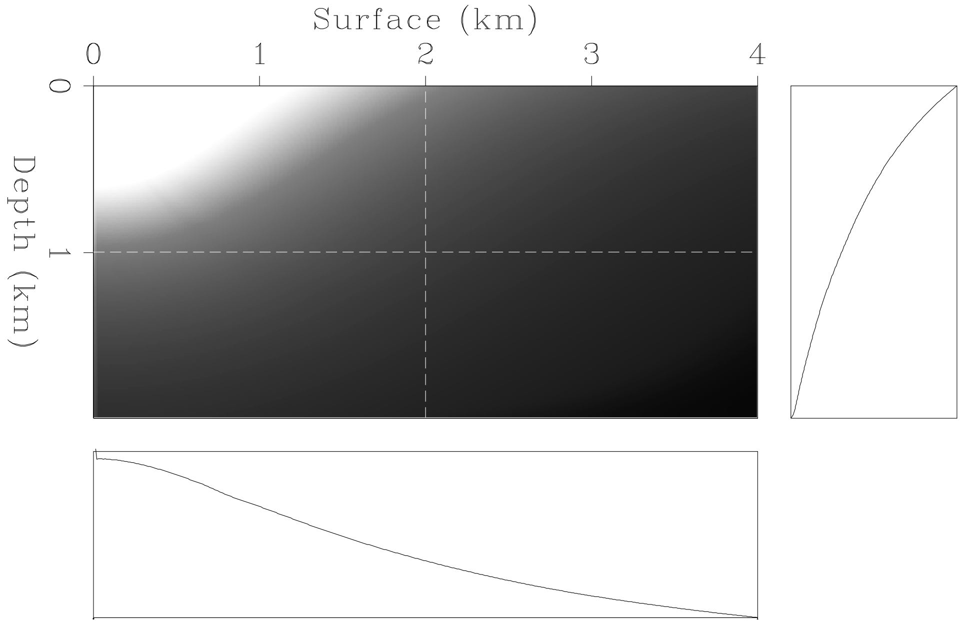

Figure 5: Amplitudes for a model with a velocity function that is linearly increasing as a function of depth. The source is a line source with an isotropic radiation pattern, located at the origin. The right and bottom panels show cross sections of the amplitudes as functions of depth and horizontal distance, respectively. The dashed lines indicate the positions of the two cross sections. lin1-ampgrals [ER]

the amplitudes computed for a line source having an isotropic radiation pattern. The computed amplitudes for a point source having an isotropic radiation pattern are shown in Figure **??**. Comparing these two figures, we see that for a point source, the amplitudes decay faster than for a line source. Both figures show that large errors appear in a narrow region just below the source position. These errors are due to the approximation of using constant velocity cells in polar coordinates. To solve this problem, I have proposed a new approach called the local paraxial ray method (Zhang, 1991). I expect that the local paraxial ray method can handle the amplitude calculation for slowness models containing reasonably large contrasts.

CONCLUSIONS

This paper presents a new algorithm for calculating the Green's function of first arrivals in two dimensions using asymptotic ray approximation. The traveltimes and amplitudes associated with the approximate Green's function are computed on a regular grid using a finite difference scheme. This algorithm is efficient and stable. Examples with different types of slowness models show that for the traveltime calculation, the algorithm can handle very sharply-contrasted slowness models, and for the amplitude calculation, the algorithm requires the slowness models to smooth. I

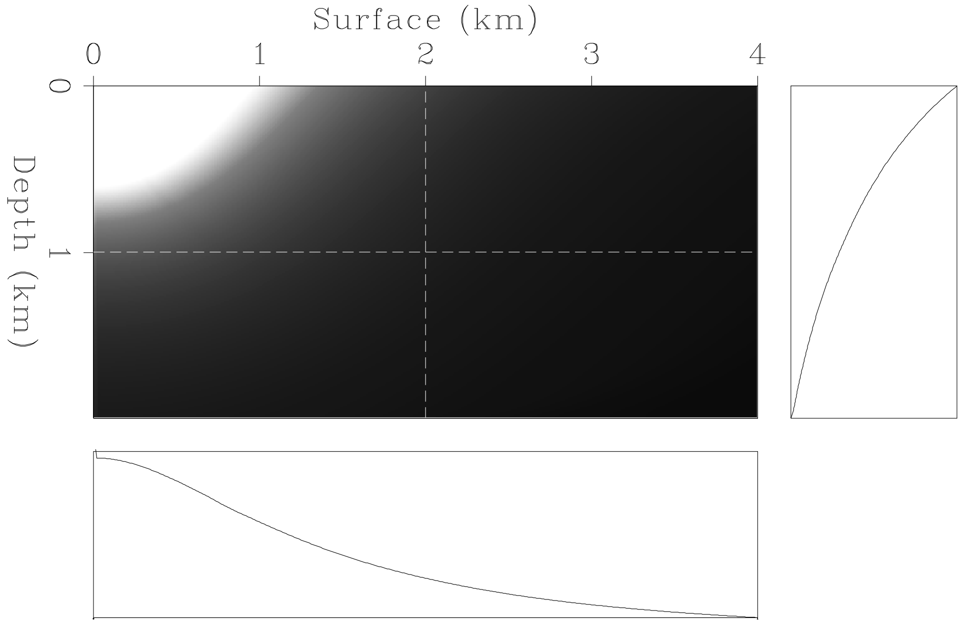

Figure 6: Amplitudes for a model with a velocity function that is linearly increasing as a function of depth. The source is a point source with an isotropic radiation pattern, located at the origin. The right and bottom panels show cross sections of the amplitudes as functions of depth and horizontal distance, respectively. The dashed lines indicate the positions of the two cross sections. lin1-ampgraps [ER]

expect that the algorithm and its extension to 3-D will be useful in Kirchhoff prestack depth migration and in traveltime tomographic inversion. I also plan to apply this algorithm to the conversion of migrated time sections to depth sections and to residual depth migration.

ACKNOWLEDGMENTS

I thank Carlos Cunha-Filho for many interesting discussions. I also thank David Lumley and Steve Cole for their proofreading of this paper.

REFERENCES

Beydoun, W.B., and Keho, T.H., 1987, The paraxial ray method: Geophysics, **52**, 1639–1653.

Červený, V., Molotkov, I.A., and Pšenčik, I., 1977, Ray methods in seismology: Univ. of Karlova Press.

Červený, V., 1987, Ray tracing algorithms in three-dimensional laterally varying layered structures: Seismic Tomography, edited by G. Nolet, Riedel Publishing Company, 99–134.

Podvin, P., and Lecomte, I., 1991, Finite difference computation of traveltimes in very contrasted velocity models: a massively parallel approach and its associated tools: Geophys. J. Int., **105**, 271–284.

Van Trier, J., and Symes, W.W., 1991, Upwind finite-difference calculation of traveltimes: Geophysics, **56**, 812–821.

Vidale, J.E., 1988, Finite-difference calculation of travel times: Bull. Seism. Soc. Am., **78**, no. 6, 2062–2076.

Vidale, J.E., and Houston, H., 1990, Rapid calculation of seismic amplitudes: Geophysics, **55**, 1504–1507.

Zhang, L., 1991, Local paraxial ray method: SEP–**72**, 171–176.

APPENDIX A

In this appendix, I derive the formula for calculating the Jacobian function $J(x,z)$ and the partial differential equation (14).

From equation (12), we have

$$J(x,z) = \begin{pmatrix} \frac{\partial x}{\partial s} & \frac{\partial z}{\partial s} \\ \frac{\partial x}{\partial \gamma} & \frac{\partial z}{\partial \gamma} \end{pmatrix} = \begin{pmatrix} \frac{\partial s}{\partial x} & \frac{\partial \gamma}{\partial x} \\ \frac{\partial s}{\partial z} & \frac{\partial \gamma}{\partial z} \end{pmatrix}^{-1} = \frac{1}{\frac{\partial s}{\partial x}\frac{\partial \gamma}{\partial z} - \frac{\partial s}{\partial z}\frac{\partial \gamma}{\partial x}}. \qquad (A.1)$$

The expression in the denominator of the last line is a cross-product of two gradient vectors:

$$\frac{\partial s}{\partial x}\frac{\partial \gamma}{\partial z} - \frac{\partial s}{\partial z}\frac{\partial \gamma}{\partial x} = \nabla s \times \nabla \gamma = \frac{1}{m(x,z)} \nabla \tau \times \nabla \gamma. \qquad (A.2)$$

$\gamma(x,z)$ is the take-off angle of the ray that reaches point(x,z), and is constant along each ray. Thus, the gradient directions of γ are always orthogonal to rays. On the other hand, the gradient directions of traveltimes are always tangential to rays. Consequently, $\nabla\tau$ and $\nabla\gamma$ are orthogonal, which yields

$$\nabla\tau \cdot \nabla\gamma = 0 \qquad (A.3)$$

and

$$|\nabla\tau \times \nabla\gamma| = |\nabla\tau||\nabla\gamma|. \qquad (A.4)$$

In 2-D, equation (A.3) can be rewritten as

$$\tau_x \gamma_x + \tau_z \gamma_z = 0 \tag{A.5}$$

which is equation (14). From equations (A.1), (A.2) and (A.4), we can derive

$$|J(x,z)| = \frac{m(x,z)}{|\nabla\tau||\nabla\gamma|}. \tag{A.6}$$

The eikonal equation states that $|\nabla\tau|^2 = m^2(x,z)$ or $|\nabla\tau| = m(x,z)$. Applying this relation to equation (A.6), we get equation (12):

$$|J(x,z)| = \frac{1}{|\nabla\gamma|} = \frac{1}{\sqrt{\left(\frac{\partial\gamma}{\partial x}\right)^2 + \left(\frac{\partial\gamma}{\partial z}\right)^2}}. \tag{A.7}$$

APPENDIX B

To derive equation (16), I take the derivative of both sides of equation (15) with respect to s and map the result to Cartesian coordinates, which yields

$$\frac{\partial\eta}{\partial x}\frac{dx}{ds} + \frac{\partial\eta}{\partial z}\frac{dz}{ds} = \frac{m(x_0,z_0)}{m(x,z)}. \tag{B.1}$$

The ray equation (6) gives

$$\begin{aligned} \frac{dx}{ds} &= \frac{\tau_x(x,z)}{m(x,z)} \\ \frac{dz}{ds} &= \frac{\tau_z(x,z)}{m(x,z)} \end{aligned} \tag{B.2}$$

Substituting these relations from equation (B.2) into equation (B.1) and multiplying both sizes by $m(x,z)$ yields equation (16):

$$\tau_x \eta_x + \tau_z \eta_z = m(x_0,z_0). \tag{B.3}$$

When (x,z) tends to (x_0,z_0), s tends to s_0 and η tends to zero, which gives the initial condition.